Tests for IceCube DOMs

Azriel Goldschmidt
Arthur Jones
Lawrence Berkeley National Labs

August 20, 2003

Contents

1 Scope and purpose of this document

The first prototypes of IceCube Digital Optical Modules Main Board (DOMMB) are expected early in January 2003. The first phase of testing will be mostly voltmeter and oscilloscope driven, looking for possible shorts or disconnected traces or parts, etc. Passed that point, more sophisticated tests need to be performed. These involve exercising the various parts on the DOMMB such as digitizers, discriminators, pulsers, etc. For these, software needs to be written. This document describes the tests that should be performed by the test software. We are aiming for pass/no-pass type of tests. However, some of the data produced in the testing procedure will need to be preserved and archived in a database for future reference. For each test, the following are listed:

- A list of the required functionality to run it
- A list of relevant previous tests
- The list of input values and controls: their meaning, default values and type
- The list of output values and their meaning/type
- The procedure of the test and the criteria for pass/no-pass (if relevant)

Temporary notes:

- A few fields are defined for each test, Name, Version, Loop count (which will have slightly different meaning in each case because it lives inside the test, pass/fail output (still need to define the case of N/A) and the value when an error occurs, error code, error string.
- Give a list of previous relevant tests.
- Notes from Bruce, speed of ATWDs 1.5 ± 0.05, baseline 100 $\pm$ 20 ???, amplitude 160 $\pm$ 4

2 Tests

2.1 ADC tests

In this test a single ADC channel (a board voltage or current) is read out multiple times and the mean, spread, maximum and minimum are compared to expectations to define a pass-no pass criteria.

Previous relevant tests

None

Required functionality

Read all ADC channels (8 channels of CS0 currently used in Jan 03 version board)

Inputs

Input Variable	Description	Type	Default and Range
adc_chip	Selects one of the ADC chips on the board (0 = CS0 and 1 = CS1)	unsignedInt	0 [0..0]
adc_channel	Selects one of the channels of the ADC chip to be read out	unsignedInt	0 [0..7]
loop_count	Number of ADC values to acquire to compute a mean and spread	unsignedInt	1000 [1..1000000]
adc_mean_counts	Mean of the ADC values distribution in count units	unsignedInt	
adc_rms_counts	Spread of the ADC values distribution in count units	unsignedInt	
adc_max_counts	Maximum ADC value read in count units	unsignedInt	
adc_min_counts	Minimum ADC value read in count units	unsignedInt	
adc_mean_mvolt_or_mamp	Mean of the ADC values distribution in millivolts or milliamp units	unsignedInt	
adc_rms_mvolt_or_mamp	Spread of the ADC values distribution in millivolts or milliamp units	unsignedInt	
adc_max_mvolt_or_mamp	Maximum ADC value read in milivolts or miliamp units	unsignedInt	
adc_min_mvolt_or_mamp	Minimum ADC value read in millivolts or milliamp units	unsignedInt	

Outputs

Output Variable	Description	Type
adc_chip	Selects one of the ADC chips on the board (0 = CS0 and 1 = CS1)	unsignedInt
adc_channel	Selects one of the channels of the ADC chip to be read out	unsignedInt
loop_count	Number of ADC values to acquire to compute a mean and spread	unsignedInt
adc_mean_counts	Mean of the ADC values distribution in count units	unsignedInt
adc_rms_counts	Spread of the ADC values distribution in count units	unsignedInt
adc_max_counts	Maximum ADC value read in count units	unsignedInt
adc_min_counts	Minimum ADC value read in count units	unsignedInt
adc_mean_mvolt_or_mamp	Mean of the ADC values distribution in millivolts or milliamp units	unsignedInt
adc_rms_mvolt_or_mamp	Spread of the ADC values distribution in millivolts or milliamp units	unsignedInt
adc_max_mvolt_or_mamp	Maximum ADC value read in milivolts or miliamp units	unsignedInt
adc_min_mvolt_or_mamp	Minimum ADC value read in millivolts or milliamp units	unsignedInt

Procedure

- Perform LOOP_COUNT readouts of the ADC from channel ADC_CHANNEL of the ADC chip (chip-select) ADC_CHIP and calculate the mean ADC_MEAN_COUNTS (keeping the running sum in a Unsigned Long and dividing by LOOP_COUNT at the end).
- Perform additional LOOP_COUNT readouts of the same ADC and save the maximum and minimum values in ADC_MAX_COUNTS and ADC_MIN_COUNTS. Also calculate the standard deviation using the previously calculated ADC_MEAN_COUNTS:

$$adc\_rms\_counts = \sqrt{\frac{1}{loop\_count - 1} \times \sum (adc_j - adc\_mean\_counts)^2}$$

- Translate the mean, standard deviation and min & max from counts to milliamp/millivolt units using the translations below, and fill the corresponding output variables:
  - Channel 0

$$adc\_xxx\_mvolt\_or\_mamp = \frac{adc\_xxx\_counts}{2}$$

- Channel 1 (no translation for pressure here)

$$adc\_xxx\_mvolt\_or\_mamp = adc\_xxx\_counts$$

- Channel 2

$$adc\_xxx\_mvolt\_or\_mamp = \frac{adc\_xxx\_count \times 2500}{4095 \times \frac{10+24.9}{10}}$$

- Channel 3

$$adc\_xxx\_mvolt\_or\_mamp = \frac{adc\_xxx\_counts}{20}$$

- Channel 4

$$adc\_xxx\_mvolt\_or\_mamp = \frac{adc\_xxx\_counts}{20}$$

- Channel 5

$$adc\_xxx\_mvolt\_or\_mamp = \frac{adc\_xxx\_counts}{20}$$

- Channel 6

$$adc\_xxx\_mvolt\_or\_mamp = \frac{adc\_xxx\_counts}{20}$$

- Channel 7

$$adc\_xxx\_mvolt\_or\_mamp = \frac{adc\_xxx\_counts}{2}$$

- Define a pass/no-pass criteria and set passed accordingly.
  - All Channels

$$passed = adc\_max\_counts < 4095 \; and \; adc\_min\_counts > 0$$

  - Channel 0

$$\begin{gathered} passed = 200 - 20 < adc\_mean\_mvolt\_or\_mamp < 200 + 20 \; and \; \cdots \\ adc\_rms\_mvolt\_or\_mamp < 20 \; and \; \cdots \\ adc\_max\_mvolt\_or\_mamp < 200 + 40 \; and \; \cdots \\ adc\_min\_mvolt\_or\_mamp > 200 - 40 \end{gathered}$$

  - Channel 2

$$\begin{gathered} passed = 2000-100 < adc\_mean\_mvolt\_or\_mamp < 2000+100 \; and \; \cdots \\ adc\_rms\_mvolt\_or\_mamp < 20 \; and \; \cdots \\ adc\_max\_mvolt\_or\_mamp < 2000 + 120 \; and \; \cdots \\ adc\_min\_mvolt\_or\_mamp > 2000 - 120 \end{gathered}$$

2.2 DAC test

2.3 Flasher Board Interface test

2.4 On-board LED Interface test

2.5 HV Interface test

2.6 PMT Signal Discriminator test

2.7 ATWD tests

Previous relevant tests

- DAC setting

Required functionality

- Set all ATWD-related DACs for both ATWD chips.
- Set SPE discriminator threshold DAC.
- FPGA: Forced launch of ATWD.
- FPGA: SPE Discriminator launch of ATWD.
- FPGA: read out single ATWD waveforms.

2.8 ATWD Pedestal Tests

In this test a number of waveforms from an ATWD chip/channel are collected to form an average pedestal waveform. The ATWD capture can be launched by either forcing it or by setting the SPE discriminator level within the noise band. The average pedestal waveform is then analyzed and a pass/no-pass is defined based on the range of pedestal values. The average pedestal waveform can be requested as an output.

Inputs

Input Variable	Description	Type	Default and Range
atwd_sampling_speed_dac	DAC setting (of a 12 Bit device) for ATWD sampling speed	unsignedInt	2000 [0..4095]
atwd_ramp_top_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Top Voltage	unsignedInt	2400 [0..4095]
atwd_ramp_bias_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Bias current	unsignedInt	350 [0..4095]
atwd_analog_ref_dac	DAC setting (of a 12 Bit device) for ATWD Analog Reference Voltage	unsignedInt	2300 [0..4095]
atwd_pedestal_dac	DAC setting (of a 12 Bit device) for ATWD Pedestal Voltage	unsignedInt	2200 [0..4095]
atwd_chip_a_or_b	Selects one of two chips (A or B) on the board:0=ATWD-A and 1=ATWD-B	unsignedInt	0 [0..1]
atwd_channel	Selects one of 3 PMT analog signal channels with different gains	unsignedInt	0 [0..2]
atwd_trig_forced_or_spe	Selects ATWD launch method: 0=forced trigger, 1=SPE discriminator trigger	unsignedInt	0 [0..1]
spe_descriminator_uvolt	SPE discriminator in microvolts (at the PMT input). 5000 = 5mV 1 SPE	unsignedInt	0 [-5000..5000]
loop_count	Number of waveforms to take and average over before analysis	unsignedInt	1000 [1..1000000]
fill_output_arrays	Fill up the output array with the average waveform. 0=do-not-fill, 1=fill-array	boolean	false [0..0]
atwd_pedestal_amplitude	Is the peak-to-peak amplitude of the averaged pedestal waveform	unsignedInt	
atwd_pedestal_pattern	Average pedestal pattern waveform in ATWD counts units.	unsignedInt	
atwd_spe_disc_threshold_dac	Average pedestal pattern waveform in ATWD counts units.	unsignedInt	

Outputs

Output Variable	Description	Type
atwd_sampling_speed_dac	DAC setting (of a 12 Bit device) for ATWD sampling speed	unsignedInt
atwd_ramp_top_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Top Voltage	unsignedInt
atwd_ramp_bias_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Bias current	unsignedInt
atwd_analog_ref_dac	DAC setting (of a 12 Bit device) for ATWD Analog Reference Voltage	unsignedInt
atwd_pedestal_dac	DAC setting (of a 12 Bit device) for ATWD Pedestal Voltage	unsignedInt
atwd_chip_a_or_b	Selects one of two chips (A or B) on the board:0=ATWD-A and 1=ATWD-B	unsignedInt
atwd_channel	Selects one of 3 PMT analog signal channels with different gains	unsignedInt
atwd_trig_forced_or_spe	Selects ATWD launch method: 0=forced trigger, 1=SPE discriminator trigger	unsignedInt
spe_descriminator_uvolt	SPE discriminator in microvolts (at the PMT input). 5000 = 5mV 1 SPE	unsignedInt
loop_count	Number of waveforms to take and average over before analysis	unsignedInt
fill_output_arrays	Fill up the output array with the average waveform. 0=do-not-fill, 1=fill-array	boolean
atwd_pedestal_amplitude	Is the peak-to-peak amplitude of the averaged pedestal waveform	unsignedInt
atwd_pedestal_pattern	Average pedestal pattern waveform in ATWD counts units.	unsignedInt
atwd_spe_disc_threshold_dac	Average pedestal pattern waveform in ATWD counts units.	unsignedInt

Procedure

- All five ATWD DAC settings are programmed.
- The ATWD trigger mask is written according to atwd_trig_forced_or_spe.
- If the SPE trigger was requested (atwd_trig_forced_or_spe=1), calculate the SPE DAC that corresponds to SPE_DISCRIMINATOR_UVOLT (see Note on SPE/MPE DAC at the end of this document) and program it.
- Take one waveform for the ATWD_CHIP_A_OR_B/ATWD_CHANNEL channel requested.

- Repeat from step D, LOOP_COUNT times, keeping a waveform that is the sum of all.
- Divide the resulting sum waveform by LOOP_COUNT to get an average waveform.
- If FILL_OUTPUT_ARRAYS=1, fill the output array with the average pedestal waveform.
- Analyze average pedestal waveform:
  - Obtain maximum and minimum value of the average pedestal waveform.
  - Set passed to pass value if ALL of the following are true
  - Minimum value of average pedestal waveform > 0
  - Maximum value of average pedestal waveform < 1023
  - Maximum-Minimum < 20 (check with MartinŠs criteria)
- Fill the ATWD_PEDESTAL_AMPLITUDE output variable with the value of Maximum-Minimum from the previous step.

2.9 ATWD Baseline test

In this test a number of waveforms from an ATWD chip/channel are collected. For each waveform an average baseline value is calculated using all the samples. The ATWD capture can be launched by either forcing it or by setting the SPE discriminator level within the noise band. The average baseline values are used to fill a histogram of this quantity that can be output on request. The mean, standard deviation and max & min of the set of average baseline values are calculated. These calculated values are used to define a pass/no-pass criterion.

Inputs

Input Variable	Description	Type	Default and Range
atwd_sampling_speed_dac	DAC setting (of a 12 Bit device) for ATWD sampling speed	unsignedInt	2000 [0..4095]
atwd_ramp_top_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Top Voltage	unsignedInt	2400 [0..4095]
atwd_ramp_bias_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Bias current	unsignedInt	350 [0..4095]
atwd_analog_ref_dac	ATWD Analog Reference Voltage Input	unsignedInt	2300 [0..4095]
atwd_pedestal_dac	DAC setting (of a 12 Bit device) for ATWD Pedestal Voltage	unsignedInt	2200 [0..4095]
atwd_chip_a_or_b	Selects one of two chips (A or B) on the board:0=ATWD-A and 1=ATWD-B	unsignedInt	0 [0..1]
atwd_channel	Selects one of 3 PMT analog signal channels with different gains	unsignedInt	0 [0..2]
atwd_trig_forced_or_spe	Selects ATWD launch method: 0=forced trigger, 1=SPE discriminator trigger	unsignedInt	0 [0..1]
spe_discriminator_uvolt	SPE discriminator in microvolts (at the PMT input). 5000 = 5mV 1 SPE	unsignedInt	0 [-5000..5000]
loop_count	Number of waveforms for baseline histogram and to compute mean, rms, etc	unsignedInt	1000 [2..100000]
fill_output_arrays	Fill up the output array with histogram of baseline. 0=do-not-fill, 1=fill-array	boolean	false [0..0]
atwd_baseline_mean	Mean of the baseline values	unsignedInt	
atwd_baseline_rms	Standard deviation of the baseline values	unsignedInt	
atwd_baseline_min	Minimum of the baseline values	unsignedInt	
atwd_baseline_max	Maximum of the baseline values	unsignedInt	
atwd_baseline_histogram	Histogram of waveform baseline values in ATWD counts units.	unsignedInt	
atwd_spe_disc_threshold_dac	ATWD discriminator threshold DAC setting	unsignedInt	

Outputs

Output Variable	Description	Type
atwd_sampling_speed_dac	DAC setting (of a 12 Bit device) for ATWD sampling speed	unsignedInt
atwd_ramp_top_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Top Voltage	unsignedInt
atwd_ramp_bias_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Bias current	unsignedInt
atwd_analog_ref_dac	ATWD Analog Reference Voltage Input	unsignedInt
atwd_pedestal_dac	DAC setting (of a 12 Bit device) for ATWD Pedestal Voltage	unsignedInt
atwd_chip_a_or_b	Selects one of two chips (A or B) on the board:0=ATWD-A and 1=ATWD-B	unsignedInt
atwd_channel	Selects one of 3 PMT analog signal channels with different gains	unsignedInt
atwd_trig_forced_or_spe	Selects ATWD launch method: 0=forced trigger, 1=SPE discriminator trigger	unsignedInt
spe_discriminator_uvolt	SPE discriminator in microvolts (at the PMT input). 5000 = 5mV 1 SPE	unsignedInt
loop_count	Number of waveforms for baseline histogram and to compute mean, rms, etc	unsignedInt
fill_output_arrays	Fill up the output array with histogram of baseline. 0=do-not-fill, 1=fill-array	boolean
atwd_baseline_mean	Mean of the baseline values	unsignedInt
atwd_baseline_rms	Standard deviation of the baseline values	unsignedInt
atwd_baseline_min	Minimum of the baseline values	unsignedInt
atwd_baseline_max	Maximum of the baseline values	unsignedInt
atwd_baseline_histogram	Histogram of waveform baseline values in ATWD counts units.	unsignedInt
atwd_spe_disc_threshold_dac	ATWD discriminator threshold DAC setting	unsignedInt

Procedure

- All five ATWD DAC settings are programmed.
- The ATWD trigger mask is written according to ATWD_TRIG_FORCED_OR_SPE.
- If the SPE trigger was requested (ATWD_TRIG_FORCED_OR_SPE=1),

calculate the SPE DAC that corresponds to SPE_DISCRIMINATOR_UVOLT (see Note on SPE/MPE DAC at the end of this document) and program it.

- Take one waveform for the ATWD_CHIP_A_OR_B/ATWD_CHANNEL channel requested.
- Calculate the baseline value: sum all samples and divide by the number of samples (128). Integer arithmetic is fine.
- Enter the value in the histogram of baseline values (it is a 1024-bins histogram).
- Keep a sum of all baseline values, and a sum of the squares for RMS calculation. Also keep the minimum and maximum value.
- Repeat from step D, LOOP_COUNT times.
- If FILL_OUTPUT_ARRAYS=1, fill the output array with the histogram of baseline values.
- Compute the mean and the RMS using the running sums (integer arithmetic is ok).
- Fill output variables ATWD_BASELINE_MEAN, ATWD_BASELINE_RMS, ATWD_BASELINE_MIN, ATWD_BASELINE_MAX.
- Set passed to pass value if ALL of the following are true:
  - ??? < ATWD_BASELINE_MEAN < ??? NEED CALCULATION OF LEVELS(WHICH DEPEND ON CHANNEL NUMBER/GAIN)
  - ATWD_BASELINE_RMS < ???
  - ATWD_BASELINE_MIN > ???
  - ATWD_BASELINE_MAX < ???

2.10 ATWD Amplitude Response

2.11 ATWD Pedestal Sweep

2.12 ATWD Cross-talk between slow control functions and ATWD PMT signal channels

2.13 ATWD Previous waveform memory

2.14 Fast ADC (FADC) test

2.15 Pressure Gauge test (work in progress need to add ADC 5V readout and translation and limits with/out temperature reading)

In this test the ADC channel corresponding to the pressure gauge is read out multiple times and the mean, spread, maximum and minimum are compared to expectations to define a pass-no pass criteria. Whether the DOMMB is in a sphere and whether it is cold or not will influence the pressure reading.

Previous relevant tests

None

Required functionality

? Read all ADC channels (8 channels of CS0 currently used in Jan 03 version board) ? If the USE_TEMPERATURE option is used requires the ability to read the on-board temperature gauge.

Inputs

Input Variable	Description	Type	Default and Range
dom_sealed	Board is sealed in sphere	boolean	True [0..0]
use_temperature	Temperature gauge reading is used in pass criteria.	boolean	False [0..0]
loop_count	Number of ADC values to acquire to compute a mean and spread	unsignedInt	1000 [0..1000000]
adc_mean_counts	Mean of the ADC values distribution in count units	unsignedInt	
adc_rms_counts	Spread of the ADC values distribution in count units	unsignedInt	
adc_max_counts	Maximum of the ADC values distribution in count units	unsignedInt	
adc_min_counts	Minimum of the ADC values distribution in count units	unsignedInt	
adc_mean_kpascal	Mean of the ADC values distribution in kPascal units	unsignedInt	
adc_rms_kpascal	Spread of the ADC values distribution in kPascal units	unsignedInt	
adc_max_kpascal	Maximum of the ADC values distribution in kPascal units	unsignedInt	
adc_min_kpascal	Minimum of the ADC values distribution in kPascal units	unsignedInt	

Outputs

Output Variable	Description	Type
dom_sealed	Board is sealed in sphere	boolean
use_temperature	Temperature gauge reading is used in pass criteria.	boolean
loop_count	Number of ADC values to acquire to compute a mean and spread	unsignedInt
adc_mean_counts	Mean of the ADC values distribution in count units	unsignedInt
adc_rms_counts	Spread of the ADC values distribution in count units	unsignedInt
adc_max_counts	Maximum of the ADC values distribution in count units	unsignedInt
adc_min_counts	Minimum of the ADC values distribution in count units	unsignedInt
adc_mean_kpascal	Mean of the ADC values distribution in kPascal units	unsignedInt
adc_rms_kpascal	Spread of the ADC values distribution in kPascal units	unsignedInt
adc_max_kpascal	Maximum of the ADC values distribution in kPascal units	unsignedInt
adc_min_kpascal	Minimum of the ADC values distribution in kPascal units	unsignedInt

Procedure

- Perform *loop_count* readouts of the ADC corresponding to the pressure gauge and of the ADC corresponding to the 5V power supply (calculate the mean *adc_mean_counts* (keeping the running sum in a Unsigned Long and dividing by *loop_count* at the end).
- Perform additional *loop_count* readouts of the same ADC and save the maximum and minimum values in *adc_max_counts* and *adc_min_counts*. Also calculate the standard deviation using the previously calculated *adc_mean_counts*:

$$adc\_rms\_counts = \sqrt{\frac{1}{(loop\_count - 1)} \times \sum (adc_j - adc\_mean\_counts)^2}$$

- Translate the mean, standard deviation and min & max from counts to kiloPascal units using the translation below and fill the corresponding output variables:

$$adc\_xxx\_kpascal = \frac{adc\_xxx\_count}{2}$$

- If *use_temperature* was set to 1, get a reading of the temperature (Units? Need to wait until I implemented the temperature test)

- Define a pass/no-pass criteria and set passed accordingly. Use the following table to determine the limits on the different output variables (pass is the logical AND of all the conditions for a given channel):

2.16 Temperature Sensor test

2.17 Analog Signal Noise test

2.18 Pulser tests

A series of important tests are to be performed with the pulser that injects PMT-like signals before the delay line and discriminator circuitry. The pulse amplitude is controlled by a DAC and it is fired by the FPGA. The pulse shape is determined by the hardware.

Required Functionality

- DAC: correct voltage output (see DAC test).
- Pulser: correct voltage output, shape and linearity as a function of DAC setting.
- FPGA: ability to continuously generate pulses at repetition rate of 1kHz. (A selectable repetition rate in the 100Hz-100kHz range would be desirable, with a default to 1kHz if not explicitly set.)

2.19 Pulser SPE/MPE discriminator test (Need to be reformatted with style of newest tests)

Required Functionality

FPGA: SPE/MPE discriminator counters. The counting gate should be about one second (preferably exactly one second). The discriminator should be allowed to fire again only after 5 microseconds. (A selectable dead-time would be desirable, which defaults to 5 microseconds if not set.)

Inputs

- Pulse height in MICROVOLTS (at the PMT input). Unsigned Long. Default = 1000 (= 1mV 0.2 SPE)
- Discriminator level in MICROVOLTS (ŞdefinedŤ at the PMT input). Signed Long. Default = 500 (= 0.5 mV 0.1 SPE)
- SPE or MPE discriminator. Boolean (0 = SPE, 1=MPE). Default = 0 (SPE).
- Pedestal level in MICROVOLTS. Unsigned Long. Default = 3,000,000 (=3V)

- Pulser repetition rate in Hz. Unsigned Integer. Default = 0 (this should choose the FPGA default of 1kHz).
- Maximum rate deviation (absolute value) from expected value in Hz. Unsigned Integer. Default = 10.

Outputs

- The measured rate in Hz. Unsigned Long.
- Test Pass/No-Pass. Boolean.

Procedure

The test proceeds as follows:

- The amplitude of the pulses is used to calculate the value of the PULSER DAC.
- The PULSER DAC is programmed with the target value calculated in the previous step.
- The PEDESTAL DAC is calculated from the input Pedestal value.
- The PEDESTAL DAC is programmed to the value from the previous step.
- The discriminator level is used to calculate the discriminator SPE/MPE DAC.
- The SPE/MPE DAC is programmed with the target value from the previous step.
- The pulser is set to fire at the nominal 1kHz rate (or different rate if Pulser repetition rate is given).
- The SPE/MPE counter is read; this step involves starting the counter and reading-out the value after the counting gate is finished.
- Check whether the rate from the SPE/MPE counter is consistent with expectations; decide pass/no-pass.
- Turn off pulser (This depends on whether subsequent tests are run from clean or not).
- A single parameter is used to determine pass/no pass criteria: an absolute deviation from the nominal expected rate. That is, for values of the pulser amplitude larger than the threshold the rate should be consistent the full rate (in the default case in the [990-1010] Hz range) . For values of the pulser smaller than the threshold the rate should be consistent with zero rate ([0-10] Hz in the default case).

Notes

Notes on DAC settings, reference voltage, SPE amplitude and discriminator settings:

- A 10-bit DAC channel sets the pulser amplitude level. Its range is [0-5V], or 4.88 mV/LSB. To compute the PULSER DAC use :

$$PULSERDAC = \frac{1}{5000000} \times \frac{pulse\_in\_microvolts \times 1024 \times (R1 + R2)}{R2}$$

  [where $R1$ and $R2$ are still not determined but should be around $\frac{R1+R2}{R2} = 10$ to cover the 0.1-100 SPE range]

- A 12-bit DAC channel sets the value of the pedestal. Its range is [0-5V]. The value is chosen to provide the ATWD with its 3V target Vref. To compute the PEDESTAL DAC use:

$$PEDESTALDAC = \frac{Pedestal\_in\_microvolts \times 4096}{5,000,000}$$

  (is this a problem for LONGS, or should I think in float calculation?).

- The nominal magnitude of a single photoelectron pulse out of the transformer is 5mV (mean). Before the discriminators there is a gain of 9.6. Therefore the amplitude is 48 mV (mean) for the SPE at the discriminator.

- A 10-bit DAC channel sets the SPE/MPE discriminator level. Its range is [0-5V], or 1.22 mV/LSB. To calculate the *spe_dac* use:

$$spe\_dac = (discriminator\_in\_microvolts \times 9.6 \times \frac{2200 + 1000}{1000} + \frac{pedestal\_dac \times 5,000,000}{4096}) \cdots \times \frac{1024}{5,000,000}$$

  To calculate the *mpe_dac* use:

$$mpe\_dac = (discriminator\_in\_microvolts \times 9.6 + \frac{pedestal\_dac \times 5,000,000}{4096}) \times \frac{1024}{5,000,000}$$

2.20 Pulser SPE/MPE discriminator scan test (Need to be reformatted with style of newest tests)

In this test the pulser is set to some amplitude and the SPE/MPE discriminator is scan and the rate at each point is recorded. It is NOT a pass/no-pass test, but the results should be recorded to a database.

Required Functionality

FPGA: SPE/MPE discriminator counters. The counting gate should be about one second (preferably exactly one second). The discriminator should be allowed to fire again only after 5 microseconds. (A selectable Şdead-timeŤ would be desirable, which defaults to 5 microseconds if not set.)

Inputs

- Pulse height in MICROVOLTS (at the PMT input). Unsigned Long. Default = 1000 (= 1mV   0.2 SPE)
- Lower value of Discriminator level to scan in MICROVOLTS (defined at the PMT input). Signed Long. Default = -1000 (= -1.0 mV   -0.2 SPE).
- Upper value of Discriminator level to scan in MICROVOLTS (defined at the PMT input). Signed Long. Default = 2000 (= 2.0 mV   -0.4 SPE).
- Step size of scan in MICROVOLTS (ŞdefinedŤ at the PMT input). Unsigned Long. Default = 0 (when this value is zero the steps are single DAC setting steps).
- SPE or MPE discriminator. Boolean (0 = SPE, 1=MPE). Default = 0 (SPE).
- Pedestal level in MICROVOLTS. Unsigned Long. Default = 3,000,000 (=3V)
- Pulser repetition rate in Hz. Unsigned Integer. Default = 0 (this should choose the FPGA default of 1kHz).

Outputs

- An array of measured rates in Hz. Array of Unsigned Long (maximum length 1024 if code protects against step size smaller that DAC granularity).

Procedure

The test proceeds as follows:

- The amplitude of the pulses is used to calculate the value of the PULSER DAC.
- The PULSER DAC is programmed with the target value calculated in the previous step.
- The PEDESTAL DAC is calculated from the input Pedestal value.
- The PEDESTAL DAC is programmed to the value from the previous step.

- The pulser is set to fire at the nominal 1kHz rate (or different rate if Pulser repetition rate is given).
- A loop over the discriminator levels between Lower and Upper with the step size requested (for the default values this should scan over 20 interesting points in the electronic noise and true discriminator region, which should take about 20 seconds). At each point, the following three steps are executed:
- Calculate the discriminator SPE/MPE DAC.
- The SPE/MPE DAC is programmed with the target value from the previous step.
- The SPE/MPE counter is read; this step involves starting the counter and reading-out the value after the counting gate is finished.
- Turn off pulser (This depends on whether subsequent tests are run from clean or not).

2.21 Pulser ATWD0/1/2 test

In this test the pulser is set to fire at the nominal rate. A waveform for a given chip/channel is taken. If the pedestal-subtracted option is requested, a waveform is taken with the pulser amplitude set to zero, and it is subtracted from the first waveform. A number of these (sometimes pedestal subtracted) waveforms are averaged. The resulting average waveform (pedestal subtracted if requested) is analyzed to check if it has the expected pulse width, height and baseline. The assumption is that the values for the ATWD DAC settings have been correctly set for this ŞlotŤ of ATWDs. Therefore, the variations in sampling speed and gain (and hopefully also baseline, in the case of non-pedestal subtracted) are small between chips and therefore a pass no pass can be defined. This test checks also the input amplifiers to the various ATWD channels.

Previous relevant tests

(These will be specified as one of the tests in this list with specific input parameters-)

- ATWD Sampling speed test
- ATWD Pedestal/baseline test
- DAC setting
- Others?

Required functionality

(Some of these could be replaced by having passed the relevant previous tests)

- Set all ATWD-related DACs for both ATWD chips.
- FPGA: Launch ATWD synchronously on pulse generation (with tunable delay in 2x clocks, defaulting to a delay similar to Delay line, if zero it should mean that the pulser fires 4 clock cycles before the ATWD).
- FPGA: read out ATWD waveforms.

Inputs

Input Variable	Description	Type	Default and Range
atwd_sampling_speed_dac	DAC setting (of a 12 Bit device) for ATWD sampling speed	unsignedInt	2000 [0..4095]
atwd_ramp_top_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Top Voltage	unsignedInt	2400 [0..4095]
atwd_ramp_bias_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Bias current	unsignedInt	350 [0..4095]
atwd_analog_ref_dac	ATWD Analog Reference Voltage Input	unsignedInt	2300 [0..4095]
atwd_pedestal_dac	DAC setting (of a 12 Bit device) for ATWD Pedestal Voltage	unsignedInt	2200 [0..4095]
atwd_chip_a_or_b	Selects one of two chips (A or B) on the board:0=ATWD-A and 1=ATWD-B	unsignedInt	0 [0..1]
atwd_channel	Selects one of 3 PMT analog signal channels with different gains	unsignedInt	0 [0..2]
atwd_trig_forced_or_spe	Selects ATWD launch method: 0=forced trigger, 1=SPE discriminator trigger	unsignedInt	0 [0..1]
pedestal_subtraction	Subtract pedestal waveform before analyzing.	boolean	True [0..0]
spe_discriminator_uvolt	SPE discriminator in microvolts (at the PMT input). 5000 = 5mV 1 SPE	unsignedInt	0 [-5000..5000]
pulse_delay_wrt_launch	Location of pulse within waveform: 2x clock cycles delay. A value of 0 is a pulse generated when ATWD launches.	unsignedInt	[0..0]
loop_count	Number of waveforms to take and average over before analysis	unsignedInt	1000 [2..100000]
atwd_waveform_baseline	Waveform baseline in ATWD counts units.	unsignedInt	
atwd_waveform_width	Pulse width in ATWD samples units	unsignedInt	
atwd_waveform_position	Pulse peak position in ATWD samples units.	unsignedInt	

Outputs

Output Variable	Description	Type
atwd_sampling_speed_dac	DAC setting (of a 12 Bit device) for ATWD sampling speed	unsignedInt
atwd_ramp_top_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Top Voltage	unsignedInt
atwd_ramp_bias_dac	DAC setting (of a 12 Bit device) for ATWD Ramp Bias current	unsignedInt
atwd_analog_ref_dac	ATWD Analog Reference Voltage Input	unsignedInt
atwd_pedestal_dac	DAC setting (of a 12 Bit device) for ATWD Pedestal Voltage	unsignedInt
atwd_chip_a_or_b	Selects one of two chips (A or B) on the board:0=ATWD-A and 1=ATWD-B	unsignedInt
atwd_channel	Selects one of 3 PMT analog signal channels with different gains	unsignedInt
atwd_trig_forced_or_spe	Selects ATWD launch method: 0=forced trigger, 1=SPE discriminator trigger	unsignedInt
pedestal_subtraction	Subtract pedestal waveform before analyzing.	boolean
spe_discriminator_uvolt	SPE discriminator in microvolts (at the PMT input). 5000 = 5mV 1 SPE	unsignedInt
pulse_delay_wrt_launch	Location of pulse within waveform: 2x clock cycles delay. A value of 0 is a pulse generated when ATWD launches.	unsignedInt
loop_count	Number of waveforms to take and average over before analysis	unsignedInt
atwd_waveform_baseline	Waveform baseline in ATWD counts units.	unsignedInt
atwd_waveform_width	Pulse width in ATWD samples units	unsignedInt
atwd_waveform_position	Pulse peak position in ATWD samples units.	unsignedInt

Procedure

The test proceeds as follows:

- All five ATWD DAC settings are programmed. M. Set the trigger masks to trigger synchronously to the pulse generation (modulo delay).
- Set the pulse delay from the PULSE_DELAY_WRT_LAUNCH input.

- Set the pulser repetition rate to the default value (writing a zero to the corresponding register).
- If pedestal subtraction is requested set the pulser height to zero microvolts.
- If pedestal subtraction is requested, take one waveform for the ATWD_CHIP_A_OR_B/ ATWD_CHANNEL channel requested.
- The PULSER_AMPLITUDE_UVOLT input is used to calculate the value of the PULSER DAC [need engineerŠs input].
- The PULSER DAC is programmed with the target value calculated in the previous step.
- Take one waveform for the ATWD_CHIP_A_OR_B/ ATWD_CHANNEL channel requested.
- If pedestal subtraction is requested, subtract the first wave from the second (result is Signed).
- Repeat from step E, LOOP_COUNT times, keeping a waveform that is the sum of all.
- Turn off pulser.
- Divide the resulting sum waveform by LOOP_COUNT to get an average waveform.
- Perform analysis on waveform to determine baseline, width, height and position.
- To calculate the baseline, make three separate averages of samples [60-79], [80-99] and [100-119] and pick the smallest of the three as the ATWD_WAVEFORM_BASELINE value.
- Find the position ATWD_WAVEFORM_POSITION and value of the waveform maximum point.
- Calculate the ATWD_WAVEFORM_HEIGHT subtracting the baseline from the maximum amplitude.
- Calculate the ATWD_WAVEFORM_WIDTH (basically a FWHM) by going to the left and right of the maximum amplitude position and finding the first sample that is smaller or equal half of the pulse height (plus the baseline). Subtract the left position from the right position.
- Check if the four values calculated in the previous step agree with expectations and set passed accordingly if ALL passed.

- The expected ATWD_WAVEFORM_HEIGHT depends on the ATWD channel (gain). Assuming that 2Volts is the entire ATWD range [0-1023] and that the gains are 8 counts/mV, 2 counts/mV and 0.3333 counts/mV for Channel 0, 1 and 2 respectively. For instance if we have PULSER_AMPLITUDE_UVOLT at 5000 and we are looking at channel 1 of the ATWD, we expect a height of 5000 * 2 mV/count / 1000 = 10 counts. The measured height should be within 20

- The expected ATWD_WAVEFORM_BASELINE depends on whether pedestal subtraction was requested. If pedestal subtraction was requested, the value should be close to zero. If it was not requested, the value would depend on the gain and pedestal/analog reference settings. Therefore the condition will be limited to: If pedestal subtraction was requested, the ATWD_WAVEFORM_BASELINE should be consistent with zero to within 2 counts.

- The expected ATWD_WAVEFORM_WIDTH depends on the true pulse width as determined from the time constants on the hardware components, the sampling speed and the bandwidth of the amplifiers. The result should be independent of all other setting and of the channel being used. The measured ATWD_WAVEFORM_WIDTH should be within one sample of the expected width [this expected value needs to be filled in by the engineers or from initial oscilloscope pulser tests, but the 3.333 nsec/sample should be used as a fixed input parameter].

- The expected ATWD_WAVEFORM_POSITION depends on the timing of pulser circuitry, the electrical length of the delay line, the sampling speed, the FPGA related delays and the requested delay between pulse and launch/trigger. The measured ATWD_WAVEFORM_POSITION should be within three samples of the expected value [this expected value needs to be filled in by the engineers or from initial measurements but it should assume 3.333 nsec/sample and discrete additional delays from PULSE_DELAY_WRT_LAUNCH of 25 nsecs/unit].

2.22 Pulser ATWD0/1/2 previous pulse memory test Into FADC

2.23 Pulser ATWD inter-channel cross-talk test

2.24 Pulser SPE spectrum with discriminator?

2.25 PMT Coupling tests

2.26 Multiplexer to ATWD channel-3 tests

2.27 Clock sinusoidal wave (sampling speed)

2.28 Clock square 2x (sampling speed)

2.29 Voltage on On-board LED

2.30 Voltage from Flasher Board Interface

2.31 Local Coincidence Signal from Upper DOM

2.32 Local Coincidence Signal from Lower DOM

2.33 Signal from Communications Input Amplifier

2.34 Pulser Signal before Delay Line

2.35 Clock square 2x ATWD inter-channel cross-talk test

2.36 Communications test

2.37 Local Coincidence Hardware test

2.38 Clock Stability test

2.39 Communications ADC Noise test

2.40 Power Consumption test

2.41 Memory tests

2.42 Single Photoelectron Spectrum test

2.43 Cross-talk within board tests

2.44 Communications signals into PMT signal path cross-talk test

2.45 Transmitted signals

Transmitted signals

2.46 Received signals

Received signals

2.47 On-board LED electrical pulsing into PMT signal path cross-talk test

2.48 Flasher board electrical pulsing into PMT signal path cross-talk test

2.49 Local coincidence signals into PMT signal path cross-talk test

2.50 PMT signal into communications signal path cross-talk test

2.51 Pulser signal into communications signal path cross-talk test

2.52 On-board LED electrical pulsing into communications signal path cross-talk test

2.53 Flasher board electrical pulsing communications signal path cross-talk test

2.54 Local coincidence signals into communications signal path cross-talk test

2.55 Reading ADCs and pressure and temperature into PMT signal path cross-talk test

2.56 Reading ADCs and pressure and temperature into communications signal path cross-talk test

2.57 Two-DOM tests

2.58 Communications cross-talk

3 Notes

- Pulser DAC: A 10-bit DAC channel sets the pulser amplitude level. Its range is [0-5V], or 4.88 mV/LSB. To compute the *pulser_dac* use :

  $$pulser\_dac = Pulse\_in\_microvolts * 1024 * (R1 + R2)/R2/5,000,000$$

  [where $R1$ and $R2$ are still not determined but should be around $\frac{R1+R2}{R2} = 10$ to cover the 0.1-100 SPE range]

- Pedestal DAC:

  A 12-bit DAC channel sets the value of the pedestal. Its range is [0-5V]. The value is chosen to provide the ATWD with its 3V target Vref. To compute the *pedestal_dac* use:

$$pedestal\_dac = \frac{pedestal\_in\_microvolts \times 4096}{5,000,000}$$

  (is this a problem for LONGS, or should I think in float calculation?).

- ADCs: The ADC (eight-channel unit) is a 12-Bit (0-4095) unit with an input range of 2.048 Volts.

- PE amplitude: The nominal magnitude of a single photoelectron pulse out of the transformer is 5mV (mean). Before the discriminators there is a gain of 9.6. Therefore the amplitude is 48 mV (mean) for the SPE at the discriminator.

- SPE/MPE DAC: A 10-bit DAC channel sets the SPE/MPE discriminator level. Its range is [0-5V], or 1.22 mV/LSB. To calculate the *spe_dac* use:

$$spe\_dac = (discriminator\_in\_microvolts \times 9.6 \times \frac{2200+1000}{1000} + \frac{pedestal\_dac \times 5,000,000}{4096}) \cdots$$
$$\times \frac{1024}{5,000,000}$$

- To calculate the *mpe_dac*

$$mpe\_dac = (discriminator\_in\_microvolts \times 9.6 + \frac{pedestal\_dac \times 5,000,000}{4096}) \times 10245,000,000$$

- Gains in the ATWD0,1,2 amplifiers: As of Jan 28 2003
  - ATWD0 has x2x4.4
  - ATWD1 has 4.333333
  - ATWD2 has x2/3
  - ATWD3
  - ADC 1x4.88x4.88